Claims

What is claimed is:

1. A system for converting data in a first hierarchical data scheme into a second hierarchical data scheme, comprising:

a template defining the second hierarchical data scheme;

a dynamic data generation module contained in the template; and

a data source, in communication with the dynamic data generation module, containing data in the first hierarchical data scheme.

2. The system of claim 1, wherein the template and the dynamic data generation module are contained in a server.

3. The system of claim 2, further including a driver connected between the dynamic data generation module and the data source.

4. The system of claim 3, further including a developer module contained in the server for creating the dynamic data generation module.

5. The system of claim 1, wherein the template is a static extensible markup language document.

6. The system of claim 1, wherein the template is an extensible markup language document type definition.

7. The system of claim 1, wherein the template is an extensible markup language schema.

8. The system of claim 1, wherein the first hierarchical data scheme is selected from the group of: extensible markup language schemes, relational databases, non-relational databases, extensible markup language databases and self describing databases.

9. The system of claim 1, wherein the second hierarchical data scheme is selected from the group of: extensible markup language schemes, relational databases, non-relational databases, extensible markup language databases and self describing databases.

10. The system of claim 1, wherein the dynamic data generation module includes a query directed to the data source.

11. The system of claim 1, wherein the dynamic data generation module includes a data mapping between the first hierarchical data scheme and the second hierarchical data scheme.

12. The system of claim 4, wherein the developer module contains a wizard that walks a user through a process of creating the dynamic data generation module.

13. A method of converting data in a first hierarchical data scheme into a second hierarchical data scheme, comprising the steps of:

a) publishing a dynamic template in a server;

b) receiving an instruction from a client at the dynamic template;

c) executing the dynamic template; and

d) when a dynamic data generation module is executed, performing a data transfer operation that converts data in the first hierarchical data scheme into the second hierarchical data scheme.

14. The method of claim 13, wherein step (a) further includes the steps of:

a1) receiving a template;

a2) determining for each element of the template if a dynamically generated data is required;

a3) when the dynamically generated data is required, receiving a data source for obtaining the dynamically generated data.

15. The method of claim 14, further including the steps of:

a4) receiving a data mapping between the first hierarchical data scheme and the second hierarchical data scheme.

16. The method of claim 15 wherein step (a4) further includes the steps of:

i) when the first hierarchical data scheme is a non-extensible markup language and the second hierarchical data scheme is a second non-extensible markup language, creating a first data mapping between the first hierarchical data scheme and an intermediate extensible markup scheme;

ii) creating a second data mapping between the intermediate extensible markup scheme and the second hierarchical data scheme.

17. The method of claim 15, further including the step of"

a5) receiving a key associated with the data mapping.

18. A method of converting data in a hierarchical data scheme into an extensible markup language scheme, comprising the steps of:

a) receiving a static extensible markup language template;

b) determining for each element of the static extensible markup language template if a datum needs to be dynamically generated;

c) when the datum needs to be dynamically generated, receiving a data source having data in the hierarchical data scheme for acquiring the datum;

d) receiving a data map between a data element in the data source and a metatag in the static extensible markup language template; and

e) repeating steps (b) through (d) for every element of the static extensible markup language template to form a dynamic data conversion program.

19. The method of claim 18, wherein step (a) further includes the step of receiving a template selected from the group including: an extensible markup language document type definition and an extensible markup language schema.

20. The method of claim 18, wherein step (a) further includes the step of:

a1) defining an input parameter.

21. The method of claim 18, wherein step (c) further includes the step of:

c1) receiving a driver.

22. The method of claim 18, wherein step (c) further includes the step of:

c1) generating a query to the data source.

23. The method of claim 18, wherein step (d) further includes the step of:

d1) receiving a screen having a list of elements from the data source and a list of metatags from the static extensible markup language template.

24. The method of claim 18, wherein step (c) further includes the step of:

c1) displaying an incomplete version of a dynamic extensible markup language template wherein a static element is shown in a first color and a dynamic element is shown in a second color.

25. The method of claim 18, further including the steps of:

e) publishing the dynamic data conversion program to a server;

f) when a query is received at the server for the dynamic data conversion program, executing the dynamic data conversion program to form an extensible markup language document.

26. A method of converting data in an extensible markup language scheme into a hierarchical data scheme, comprising the steps of:

a) receiving a sample extensible markup language file;

b) determining for each element of the sample extensible markup language file if a datum needs to be dynamically processed;

c) when the datum needs to be dynamically processed, receiving an extensible markup language element location for acquiring the datum;

d) receiving a data map between a metatag in the sample extensible markup language file and an element of the hierarchical data scheme; and

e) repeating steps (b) through (d) for every element of the sample extensible markup file to form a dynamic data conversion program.

27. The method of claim 26, wherein step (a) further includes the step of:

a1) defining a key.

28. The method of claim 26, wherein step (d) further includes the steps of:

d1) receiving a query type;

d2) generating a query.

29. The method of claim 28, wherein step (d1) further includes receiving an insert query type.